A brief git introduction

Topics

1. GitHub's SSH key
2. First Steps
3. How to create a new Repository
4. How to clone (copy) a Repository
5. How to update file changes
6. How to merge conflicts
7. Further Resources

1. GitHub SSH key

GitHub is not using a password-based authentication anymore. Instead, to access and write data to a repository, GitHub uses SSH (Secure Shell Protocol). For this, a local private key file is needed for authentication.

1.1 Generating a new SSH key

For SSH key generation, paste the following in your terminal with your GitHub related mail address.

```
ssh-keygen -t ed25519 -C "your_github_email@example.com"
```

When you're prompted to "Enter a file in which to save the key", you can press **Enter** to accept the default file location.

```
> Enter a file in which to save the key (/home/YOU/.ssh/ALGORITHM):[Press enter]
```

At the prompt, that asks for a passphrase, just pressing **Enter** is enough.

```
> Enter passphrase (empty for no passphrase): [Type a passphrase]
> Enter same passphrase again: [Type passphrase again]
```

Finally, the file `id_ed25519.pub` should be created in `~/.ssh`.

2.1 Adding SHH key to GitHub

1. Go to GitHub.
2. In the upper-right corner of any page, click your profile photo, then click **Settings**.
3. In the "Access" section of the sidebar, click **SSH and GPG keys**.
4. Click **New SSH key** or **Add SSH key**.
5. In the "Title" field, add a descriptive label for the new key. For example, if you're using a personal laptop, you might call this key "Personal laptop".
6. Select the type of key, either authentication or signing. (Authentication should be used in normal use case)
7. Open the file `~/.ssh/id_ed25519.pub` on your local machine and copy its content in the "key" field.
8. Click **Add SSH key**.
9. If prompted, confirm access to your account on GitHub.

2. First Steps

When you use Git on a new computer for the first time, you need to configure a few things.
On a command line, Git commands are written as `git verb options`, where `verb` is what we actually want to do and `options` is additional optional information which may be needed for the `verb`. So to set up your setup git on your computer, first specify name and email:

```
git config --global user.name "Your Name"
git config --global user.email "your@mail.com"
```

3. How to create a new Repository

First, create a folder or go into an existing folder you like to convert to a repository (none of your data will be deleted). Here, an example of an empty folder:

```
touch README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/<username>/<repository>.git
git push -u origin master
```

To summarize:

1. The first command creates an empty `README.md` file.
2. Then second, then initialized the git repository.
3. However, at this moment, `README.md` is not a part of the repository. Thus, the file is added by `git add README.md`. You can also add all file changes in a repository by

```
git add .
```

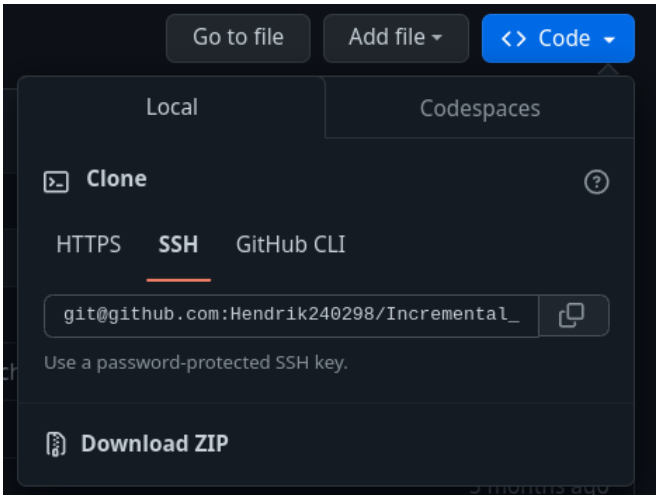
4. With the `git commit` command, you describe the changes you have made.
5. The repository is added. This only needs to be done once.
6. With the `git push` command, you push your changes to GitHub.

NOTE: With GitHub, you could also create a repository online on their website and then just clone the repository to your local machine.

4. How to clone (copy) a Repository

1. Go to the desired repository on GitHub.
2. Click on **Code** such that a pop-up opens.

3. Copy the repository link of the pop-up.



4. Open a Terminal and go to the directory, where the repository should live. Then, execute the following command, where you use the link you obtained before.

```
git clone git@github.com:<username>/<repository>.git
```

5. How to update file changes

If you have modified your code, all these changes are only locally. Your GitHub repository is not affected until now. To update the code also on GitHub and thus share your progress with your collaborators (or just for you and your version control), you need to perform the following steps.

You can specify the file(s) and/or folder(s) to update

```
git add [<file|folder>[<file|folder>[...]]]
```

Or you can add the whole directory

```
git add .
```

Now *commit* the changes (record them locally):

```
git commit -m "my commit message"
```

After one or many commits, you can use `push` to upload them into the remote repository

```
git push
```

NOTE: Many editors also provide their tools to simplify the git experience. We are using VS Code and it works perfectly and is straightforward to handle.

5.1 How to ignore files

There is often the case, where you have files in your git folder that you do not want to track. Typically, these are simulation data or temporary files. So with each `git add .` you would again add the files to your git. To prevent this, you can create a `.gitignore` file, in which you can add each file or type of file you want to neglect/ignore. E.g., you want to neglect a test code `test.py` and also your simulation data which is stored as `.npz` over several files. This can be achieved by adding the following lines to your `.gitignore`

```
test.py
*.npz
```

The `*.npz` states that all files which file name ends with `.npz` are neglected.

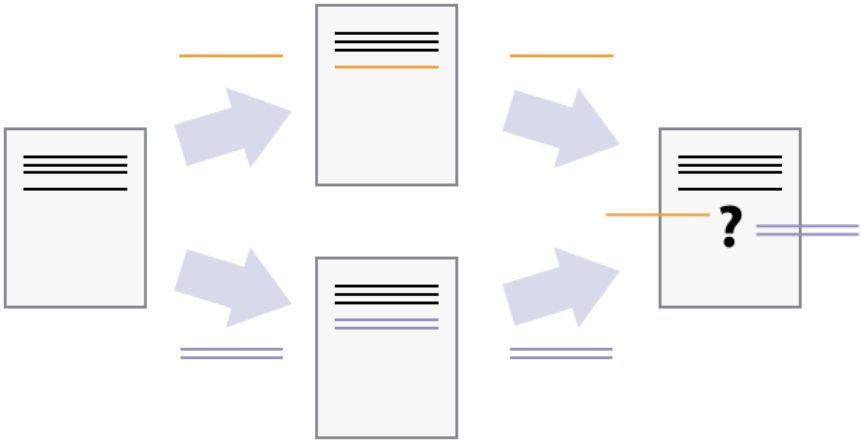
Note: It can also be useful to push your `.gitignore` such that your collaborators ignore the same files.

More details can be found here.

6. How to merge conflicts

The following is mainly based on this.

As soon as people can work in parallel, they'll likely step on each other's toes. This will even happen with a single person: if we are working on a piece of software on both our laptop and a server in the lab, we could make different changes to each copy. Version control helps us manage these conflicts by giving us tools to resolve overlapping changes.



If you try to push your changes to a repository that in the meanwhile has been changed online by another collaborator, you may get the following error message

```
To https://github.com/name/repo.git
 ! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/name/repo.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Git rejects the push because it detects that the remote repository has new updates that have not been incorporated into the local branch. What we have to do is pull the changes from GitHub, merge them into the copy we're currently working in, and then push that. Let's start by pulling:

```
git pull
```

The `git pull` command updates the local repository to include those changes already included in the remote repository. After the changes from remote branch have been fetched, Git detects that changes made to the local copy overlap with those made to the remote repository, and therefore refuses to merge the two versions to stop us from trampling on our previous work. The conflict is marked in the affected file:

```
<<<<<<< HEAD
YOUR changes
=======
Online version
>>>>>>> dabb4c8c450e8475aee9b14b4383acc99f42af1d
```

Our change is preceded by `<<<<<<< HEAD`. Git has then inserted `=======` as a separator between the conflicting changes and marked the end of the content downloaded from GitHub with `>>>>>>>`.

It is now up to you to edit this file to remove these markers and reconcile the changes. We can do anything we want: keep the change made in the local repository, keep the change made in the remote repository, write something new to replace both, or get rid of the change entirely.
NOTE: Again, tools like *VS Code* can help you to deal with merge conflicts.

To finish merging, we add the changes being made by the merge and then commit:

```
git add .
git commit -m "Merge changes from GitHub"
git push
```

6.1 Just overwrite local with GitHub version

This may not the elegant way, but if you just want the latest online version, you can force pulling.

```
git fetch --all
git reset --hard
git pull
```

Further Resources

- Top 20 Git Commands With Examples
- Git for Humans
- Git Novice
- W3schools- Getting started